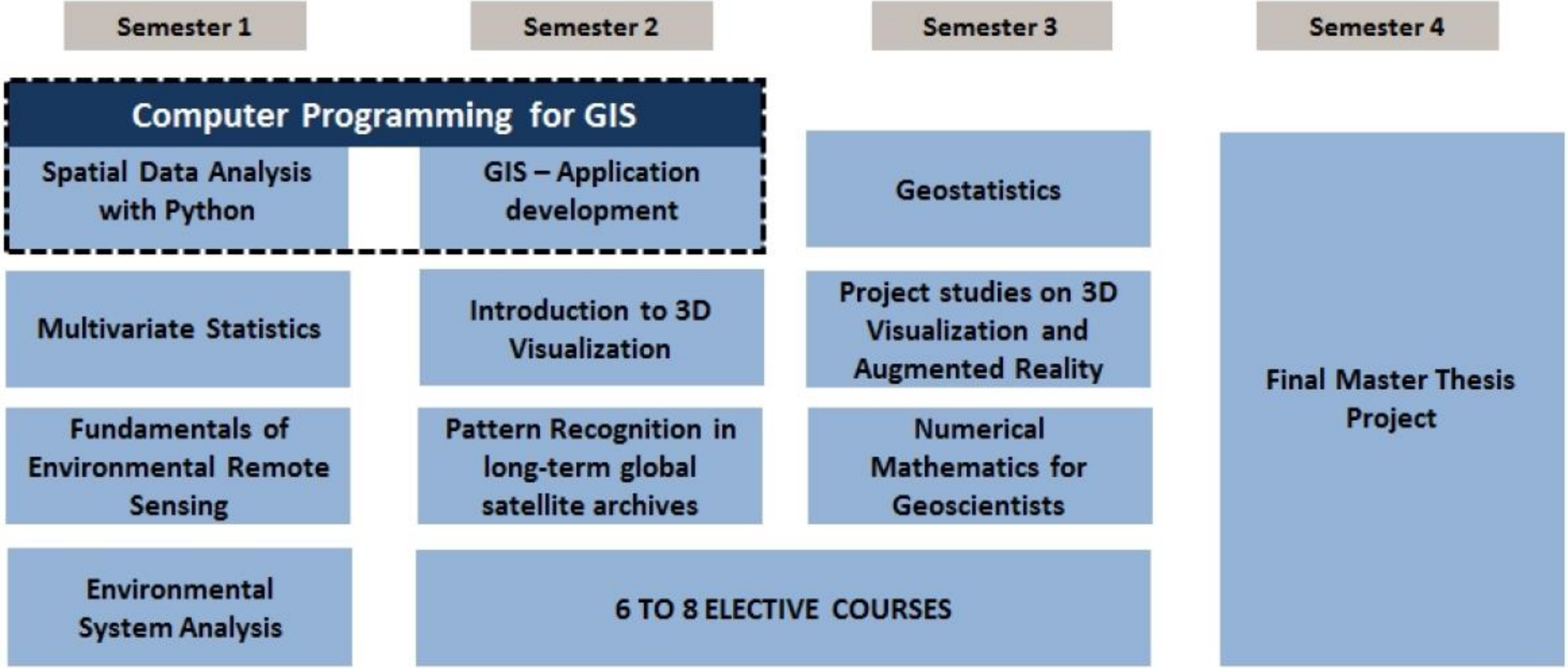
Clustering spatiotemporal point data to visualize spatial patterns

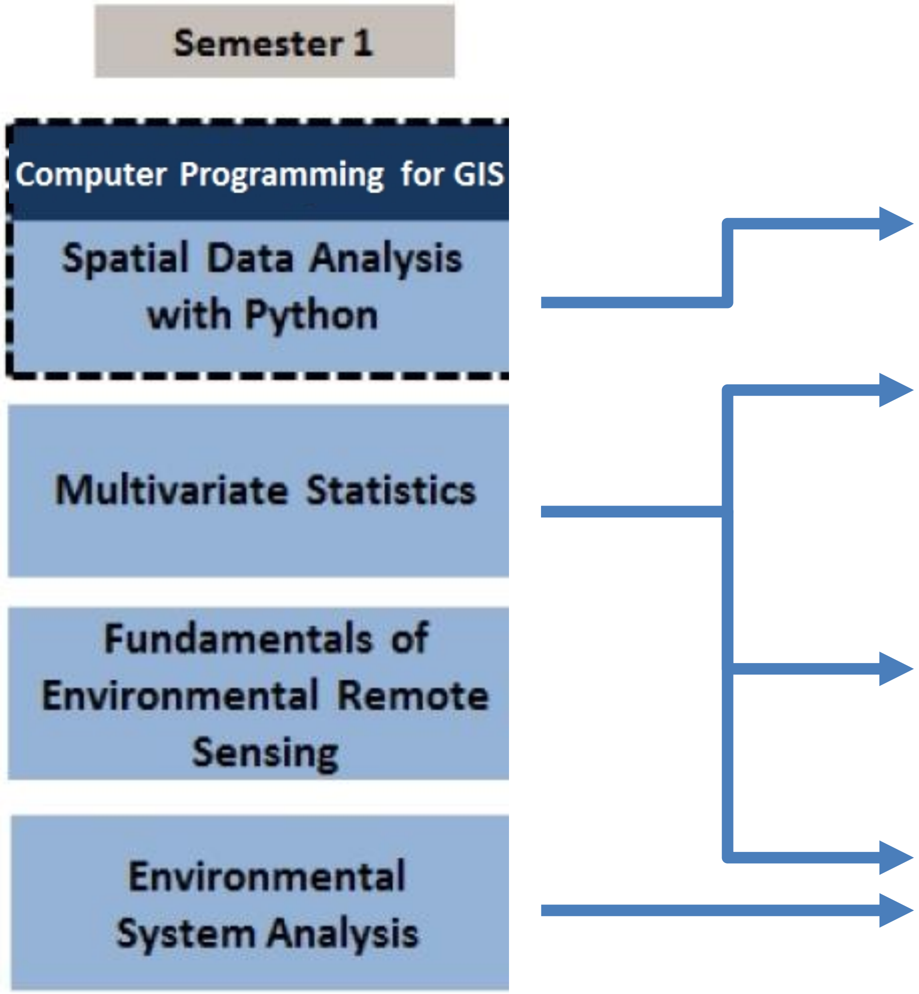
Dr. David Frantz

Geoinformatics – Spatial Data Science
Demonstration lecture
Trier / Zoom, 02.07.2020

MSc Applied Geoinformatics, 2nd term


Semester 1
Semester 2
Semester 3
Semester 4
Computer Programming for GIS
Spatial Data Analysis with Python
GIS – Application development
Geostatistics
Final Master Thesis Project
Multivariate Statistics
Introduction to 3D Visualization
Project studies on 3D Visualization and Augmented Reality
Fundamentals of Environmental Remote Sensing
Pattern Recognition in long-term global satellite archives
Numerical Mathematics for Geoscientists
Environmental System Analysis
6 TO 8 ELECTIVE COURSES

Requirements



Knowledge of geospatial data analysis

Knowledge of multivariate data and statistics

some *R* skills

Cluster analysis, concept and algorithms

Learning Objective

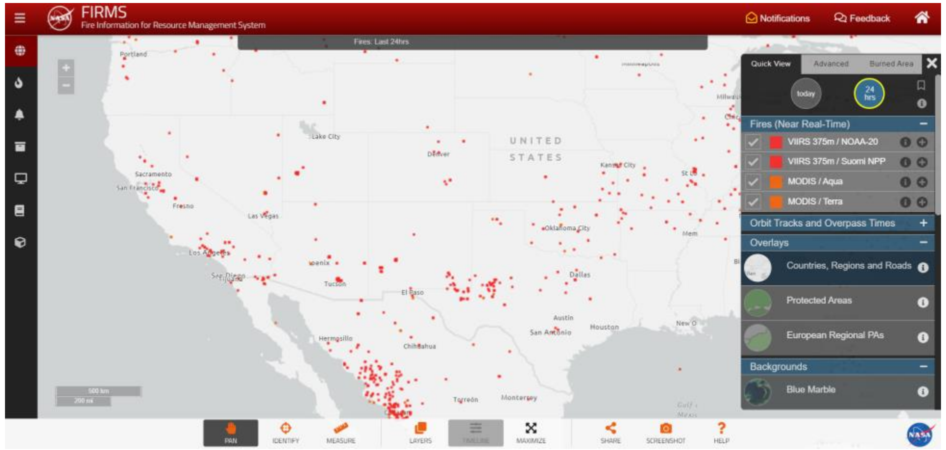
- Introduction to spatiotemporal data types
- Clustering algorithm
- Practical experience/demonstration to cluster real-life ST data with current relevancy

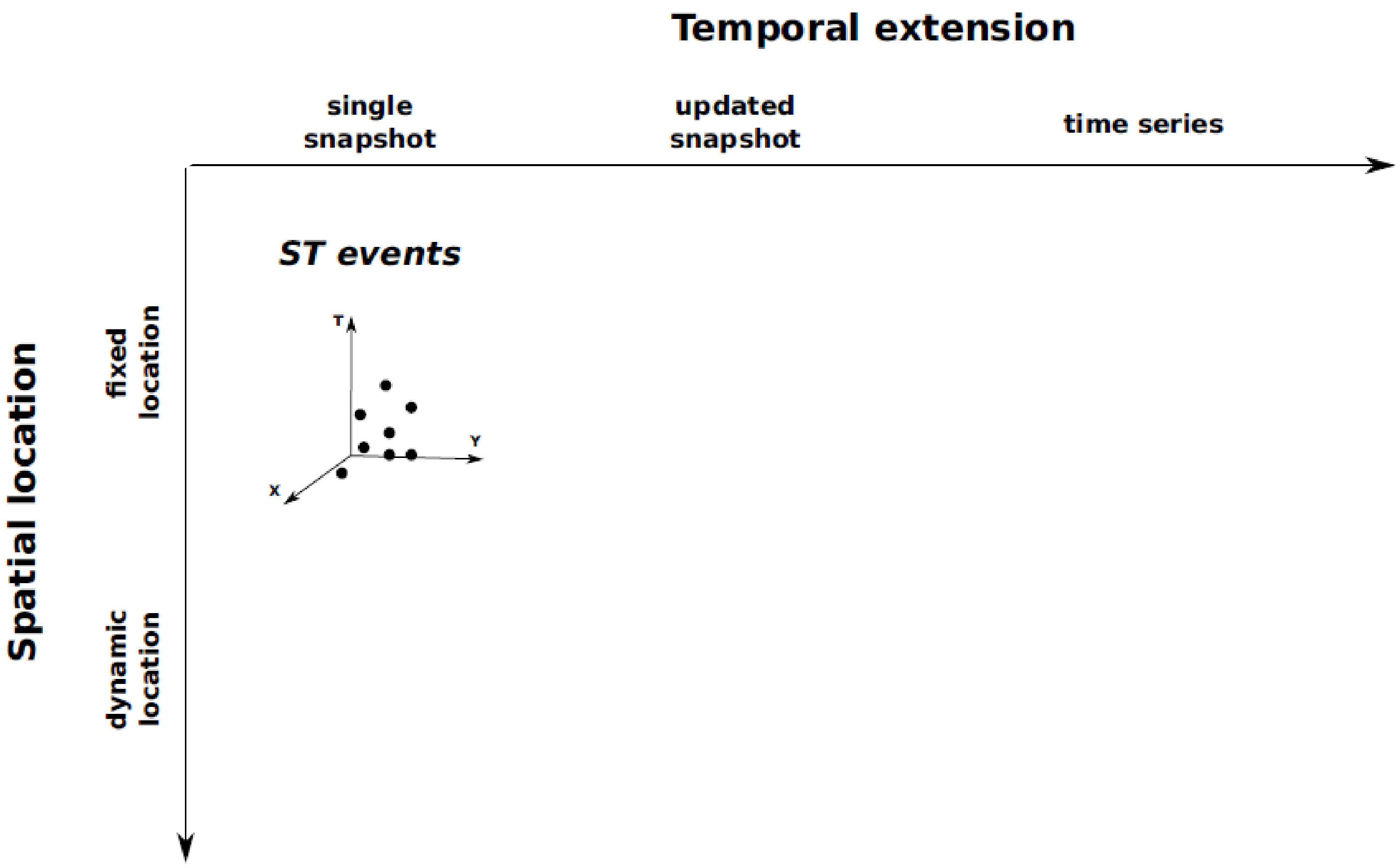
Spatiotemporal data

1) ST event

- Single measurement
- <longitude, latitude, timestamp>

Fire events





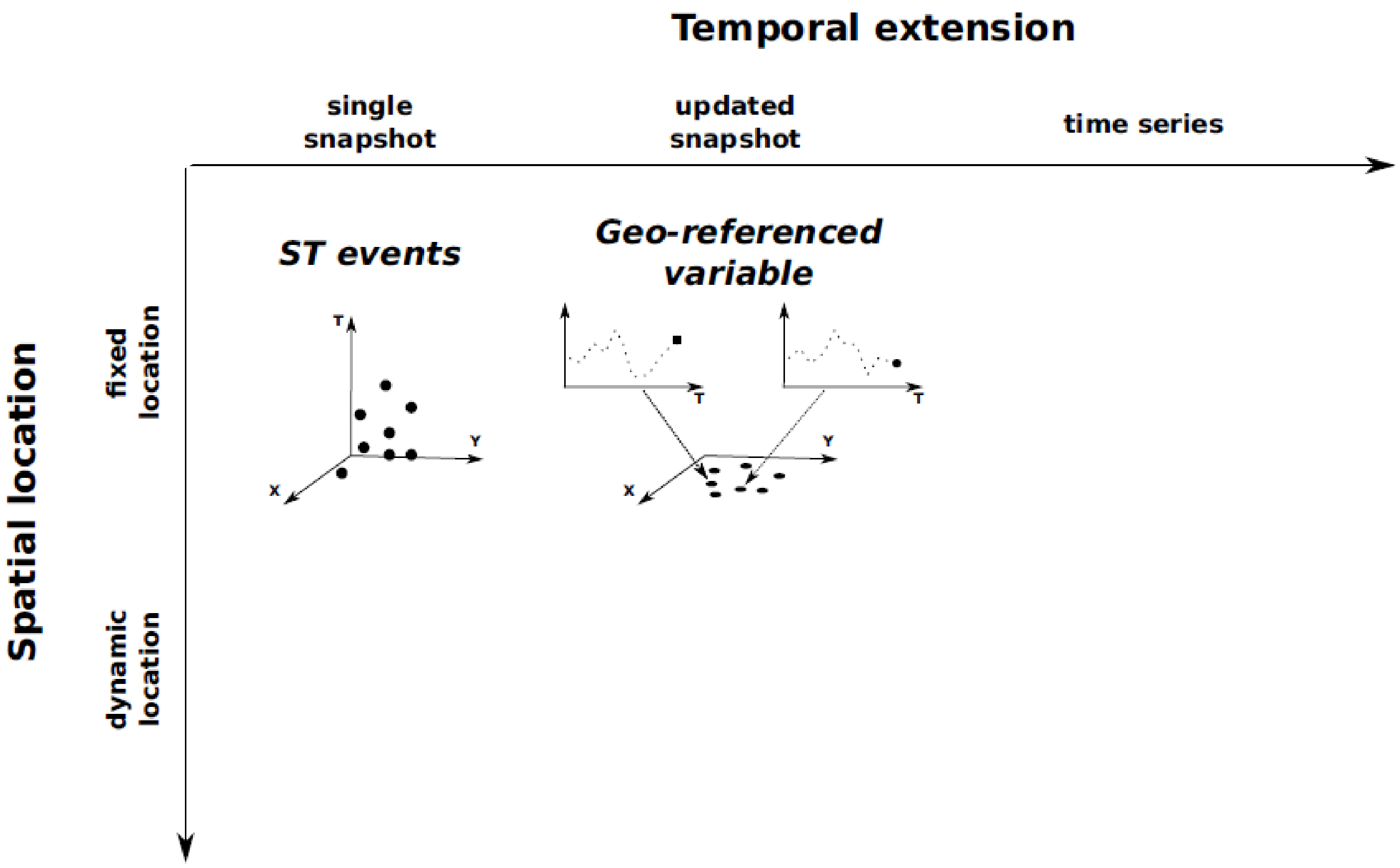
Kisilevich et al.: Spatio-temporal clustering. In: *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2009. S. 855-874.

Spatiotemporal data

2) Geo-referenced variable

- Evolution in time, but only the most recent value
- <longitude, latitude, timestamp, non-spatial value>

Weather station with
most recent temperature value



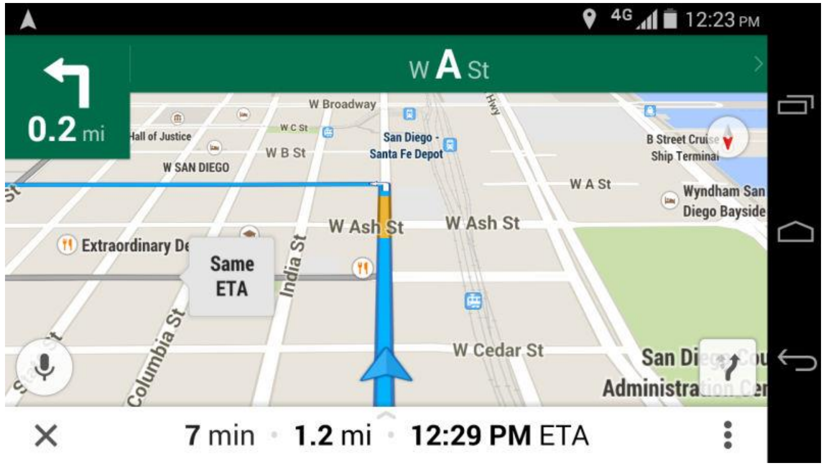
Kisilevich et al.: Spatio-temporal clustering. In: *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2009. S. 855-874.

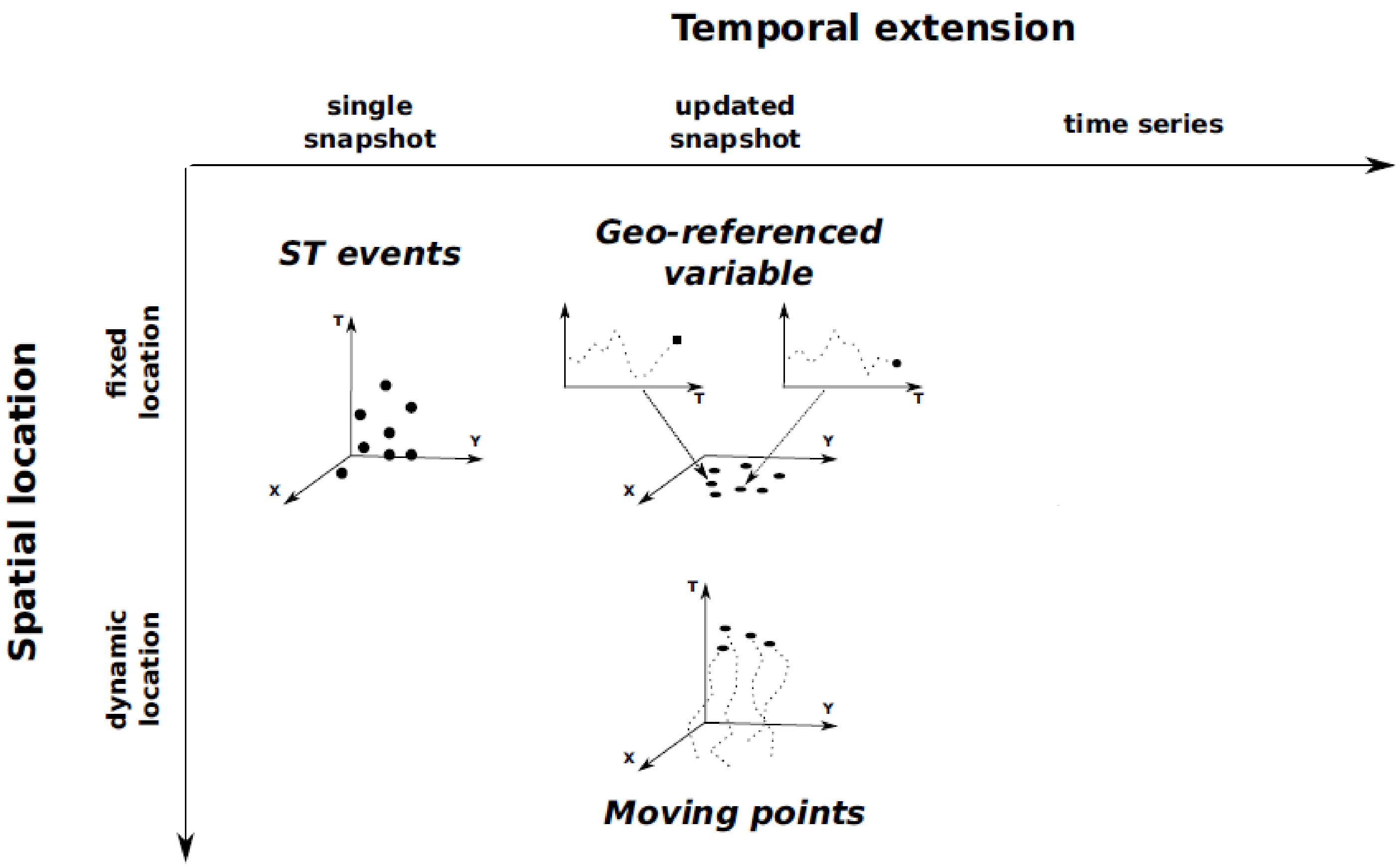
Spatiotemporal data

3) Moving points

- object moves, most recent position

navigation /
real-time tracking of vehicles





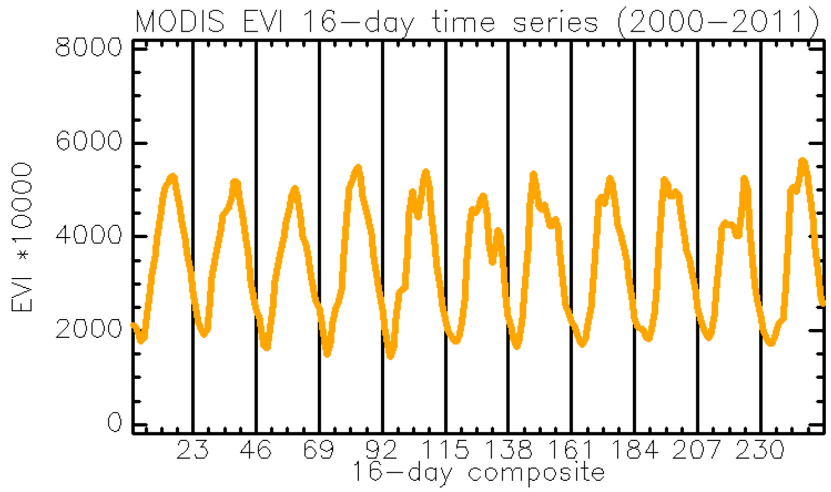
Kisilevich et al.: Spatio-temporal clustering. In:
Data mining and knowledge discovery handbook.
Springer, Boston, MA, 2009. S. 855-874.

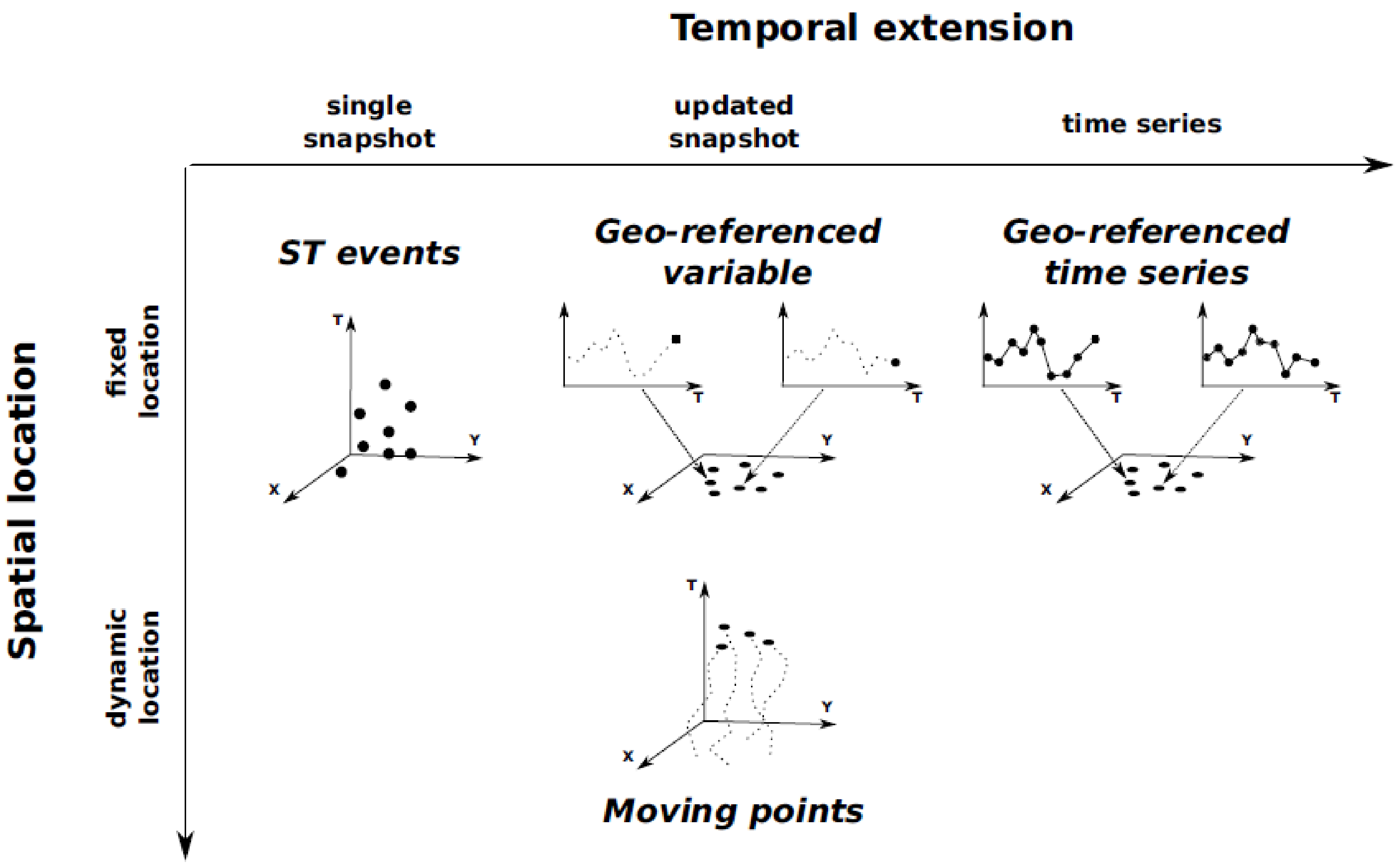
Spatiotemporal data

4) Geo-referenced time series

- Whole history is stored

NDVI time series





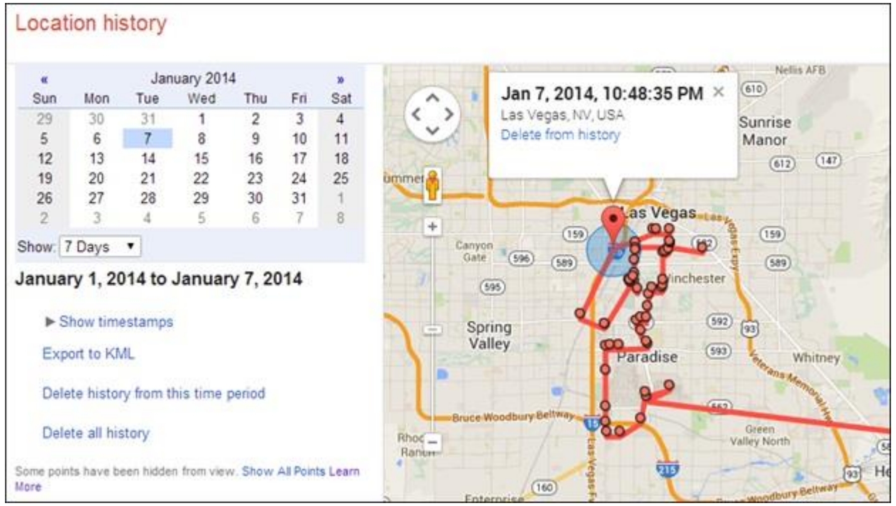
Kisilevich et al.: Spatio-temporal clustering. In: *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2009. S. 855-874.

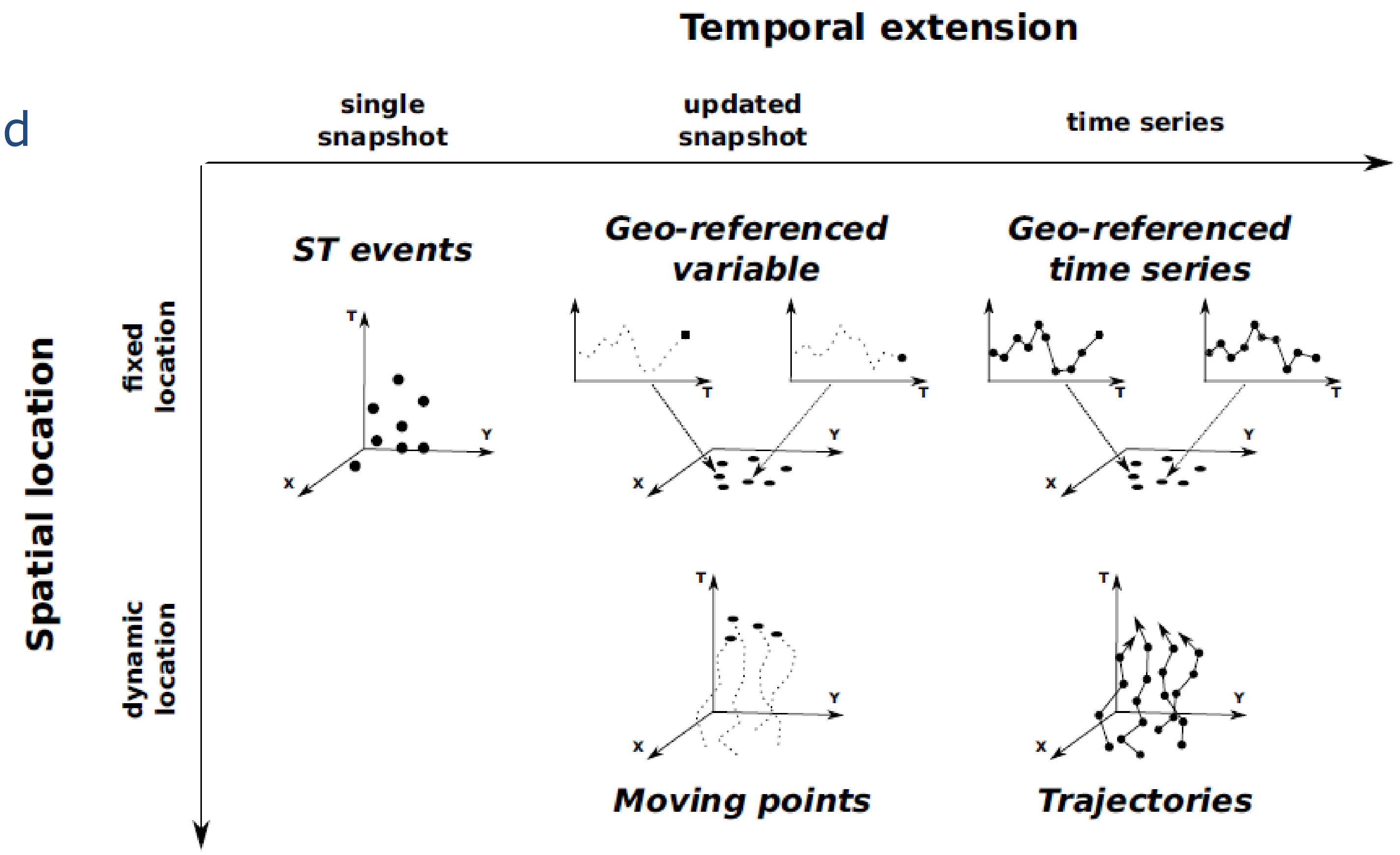
Spatiotemporal data

5) Trajectories

- Object moves, whole history is stored

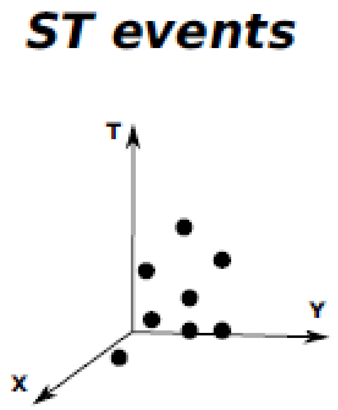
Google Location History





Kisilevich et al.: Spatio-temporal clustering. In: *Data mining and knowledge discovery handbook*. Springer, Boston, MA, 2009. S. 855-874.

Clustering ST event data



Three dimensions:
<longitude, latitude, timestamp>

Static in space and time = snapshot

Problem: complex datasets
Solution: Spatiotemporal analyses methods to mine meaningful patterns for better understanding

Clustering = unsupervised method for discovering potential patterns

Finding clusters among events means to discover groups that lie close both in time and in space

DBSCAN

Density-Based Spatial Clustering of Applications with Noise

ESTER, Martin, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In: *Kdd*. 1996. S. 226-231.

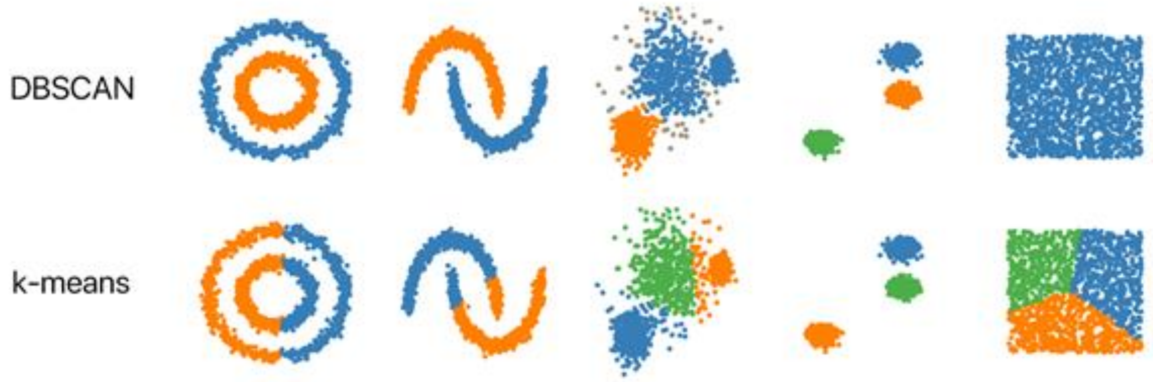
Popular algorithm in data mining, simple application, very efficient

Main assumption
Within each cluster, there is a typical density of points, which is considerably higher than outside

Find clusters of arbitrary shape

Detect noise

Number of clusters not known *à priori*



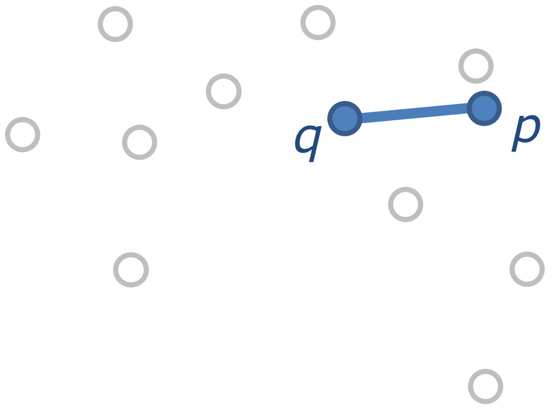
https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

DBSCAN concepts

1) Neighborhood

Determined by a distance function, e.g. Euclidean Distance

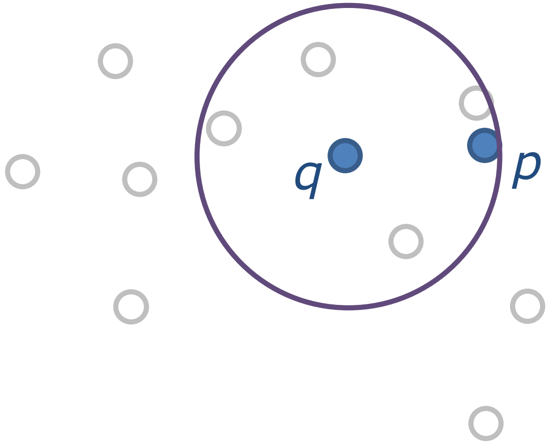
Distance between two points p and q in database D: $dist(p,q) = \sqrt{(x_p - x_q)^2 + (y_p - y_q)^2}$



DBSCAN concepts

2) Eps-neighborhood of a point q:

$$N_{Eps}(q) = \{p \in D \mid dist(p, q) \leq Eps\}$$



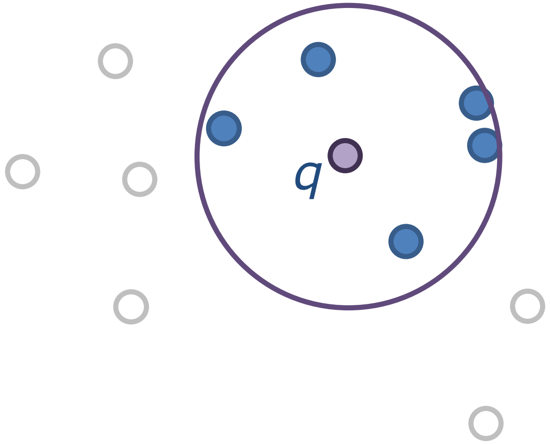
Input parameter 1:
Distance threshold *Eps*

DBSCAN concepts

3) Core point

$$\left|N_{Eps}(q)\right| \geq MinPts$$

Core point is part of a cluster



Input parameter 2:
MinPts = 3

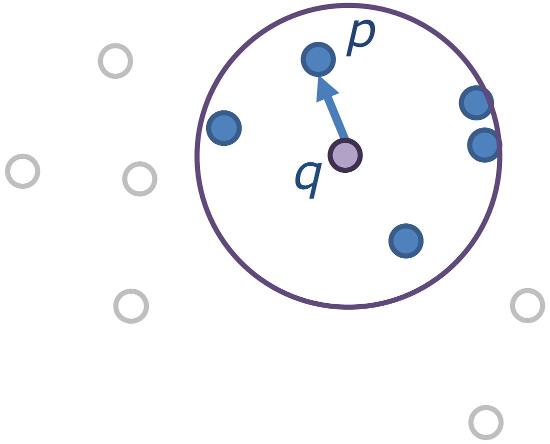
DBSCAN concepts

4) Directly density-reachable

p is directly density-reachable from q if p is within the Eps-neighborhood of q, and q is a core point

$p \in N_{Eps}(q)$ AND

$|N_{Eps}(q)| \geq MinPts$



p directly density-reachable from q

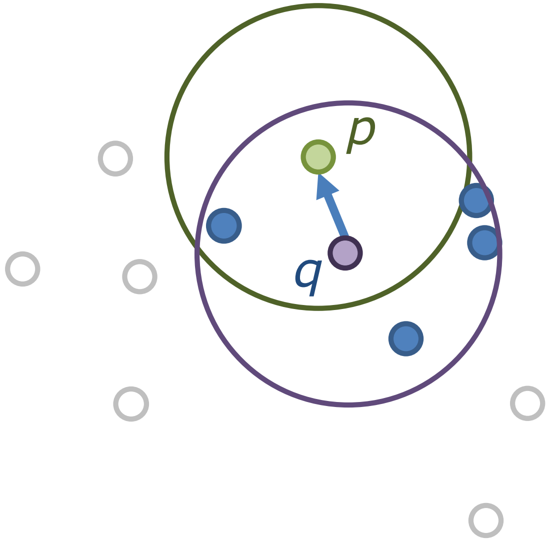
DBSCAN concepts

4) Directly density-reachable

p is directly density-reachable from q if p is within the Eps-neighborhood of q, and q is a core point

$p \in N_{Eps}(q)$ AND

$|N_{Eps}(q)| \geq MinPts$



p directly density-reachable from q

q not directly density-reachable from p

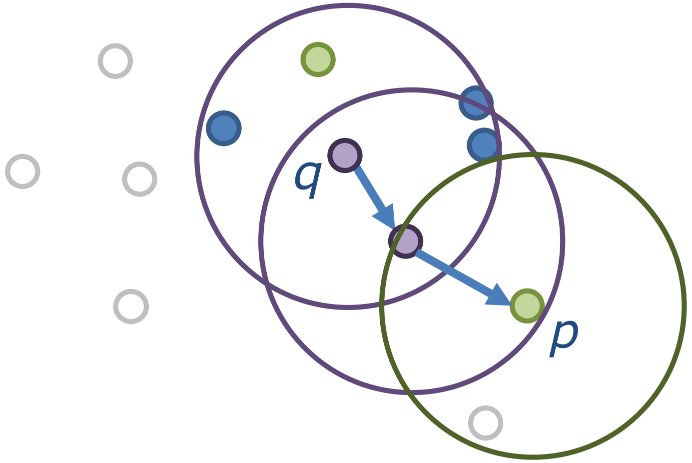
p is not a core point ($|N_{Eps}(p)|$ = 2)

→ p = **border point**

DBSCAN concepts

5) Density-reachable

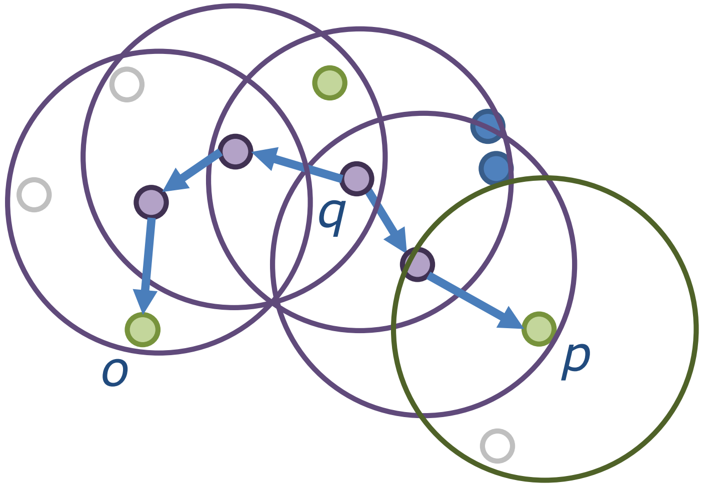
p is density-reachable from *q* if there is a chain of points that are directly density-reachable



DBSCAN concepts

6) Density-connected

p is density connected to *o*, if both *p* and *o* are density-reachable from a point *q*

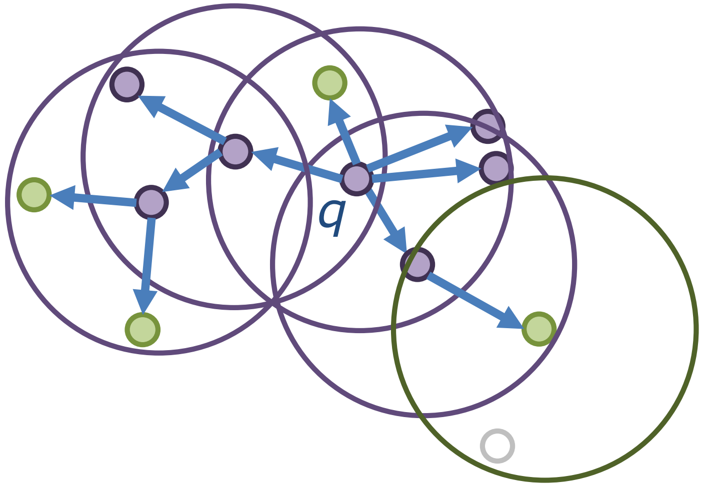


DBSCAN concepts

7) Density-based cluster contains all points that are density-reachable from a seed point q:

$\forall\, p, q : if\;\; q \in C$ AND p is density-reachable from q

$\forall\, p, q \in C : if\;\; p$ is density-connected to q



DBSCAN concepts

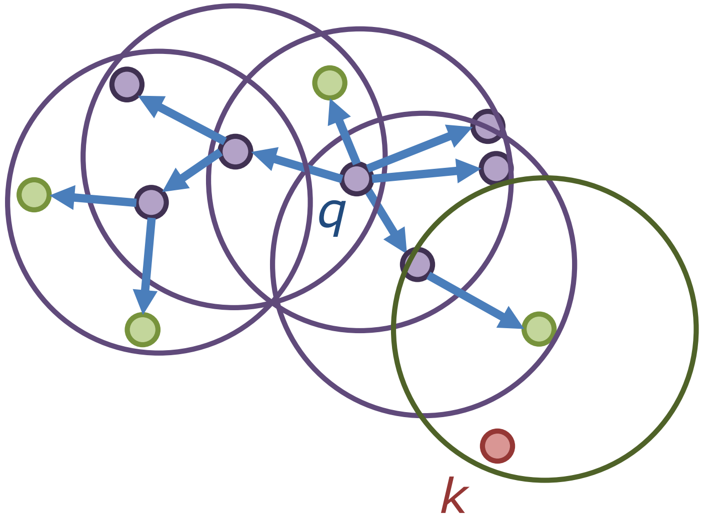
7) Density-based cluster contains all points that are density-reachable from a seed point q:

$\forall\, p, q: if\ q \in C$ AND p is density-reachable from q

$\forall\, p, q \in C: if\ p$ is density-connected to q

Noise

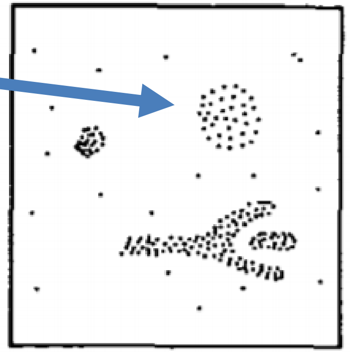
Any point k not belonging to any cluster



Eps and MinPts

MinPts does not critically affect clustering results
Suggestion use 4 for spatial data

The distance *Eps* should be set according to the “thinnest” cluster



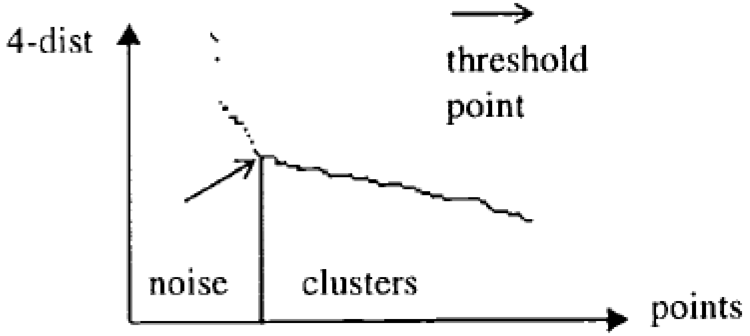
database 3

Eps and MinPts

MinPts does not critically affect clustering results
Suggestion use 4 for spatial data

The distance *Eps* should be set according to the "thinnest" cluster

Simple solution:
1) Compute the distance of a point p to its k-th nearest neighbor
    k = MinPts
2) Repeat for each point
3) Sort the distances and plot (*k-dist graph)*



Time in DBSCAN

DBSCAN can be applied to 2D, 3D or any high dimensional feature space

Time is simply an additional dimension:

$$dist(p,q) = \sqrt{\left(x_p - x_q\right)^2 + \left(y_p - y_q\right)^2 + (t_p - t_q)^2}$$

→ **some sort of scaling might be required to use the same *Eps* for space AND time**

→ **MinPts = number of dimensions + 1**

Hands-on / Live Demo

→ covid19.ipynb

Play with the data

Download the JupyterLab environment from

github.com/davidfrantz/covid19

includes
- Jupyter notebooks with all plots and code,
- COVID-19 data,
- this presentation,
- literature with suggested reading

requires
- JupyterLab
- R & R-Kernel

Parameters that will affect the clusters

- Number of infections N
  → find larger or smaller hotspots,
- Scaling of the temporal dimension
  → 7 days, 31 days?
  → statistical rescaling method for
    all dimensions? (e.g. z-transform)
- Eps
  → Shift the allocations to noise/clusters